Semester–III

S.No.	Course Code	Course Title	L	P	Credits
1	CSE210C	Data Structures	3	0	3
2	CSE211C	Object Oriented Programming	3	0	3
3	CSE212C	Database Management System	3	0	3
4	CSE213C	Discrete Mathematics	3	0	3
5	CSE214C	Digital Electronics and Logic Design	3	0	3
6	CSE215C	Data Structures Lab	0	2	1
7	CSE216C	Object Oriented Programming Lab	0	2	1
8	CSE217C	Database Management System Lab	0	2	1
9	CE218C	Digital Electronics and Logic Design Lab	0	2	1
10	CSE210A	Seminar	0	0	**-**
Total			**18**	**8**	**19**

Course Code: **CSE210C**	Course Title: **Data Structures**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Define, understand, describe and implement linear data structures like stack, queues, linked lists and nonlinear data structures like trees and graphs.
- Design and trace the algorithms for various operations on different data structures studied.
- Understand and implement various searching and sorting techniques
- Write programs in C to simulate operations and applications of data structures learnt

Unit – I (9)

Introduction to data structures, classification of Data Structures, Primitive vs. Non-Primitive data structures, Linear vs Non-Linear data structures, Primitive Data Structures Operations, Recursion Function and its Examples. String Representation and operations, Arrays representation, implementation and limitations**.**

Unit – II (10)

Linked List: Linked List and its comparison with array implementation, Singly, Doubly and Circular linked list, their Implementation and Comparison.
Stacks: Static and Dynamic Implementation. Applications of Stacks. Prefix Postfix and Infix Expressions. Infix to postfix conversion.
Queues: Static and Dynamic Implementation. Applications of Queues, Types of Queues, Array Implementation of Circular Queues.

Unit – III (9)

Searching: Sequential and Binary Search on Array-based Ordered Lists, Binary Trees, their Implementation and Traversal, Binary Search Trees: Searching, Insertion and Deletion of Nodes, Height Balanced Trees and Concept of AVL Trees, Concept and purpose of B Trees and B+ Trees.

Unit – IV (9)

Graphs and their Representations, Graph Traversal Techniques: Breadth First Search (BFS) and Depth First Search (DFS), Applications of BFS and DFS, Minimum Spanning Trees (MST), Prim's and Kruskal's algorithms for MST, Connected Components, Dijkstra's Algorithm for Single Source Shortest Paths.

Unit – V (8)

Sorting Techniques: Insertion Sort, Selection Sort, Merge Sort, Quick Sort, Heap Sort, Shell Sort, Radix Sort, Bucket sort. Concept of Hash Functions, Hash-tables and Hashing with Chaining. File Structure: Sequential Files, Indexed Files, Direct Files.

Textbooks:

1. Shaum's outlines "*Data Structures with C*" Seymour Lipschutz, Tata McGraw Hill Education.
2. Langsam Augenstein Tenenbaum "*Data Structures using C and C++"*

Reference Books:

1. Data Structures and Algorithms by Narsimha Kamarachi.
2. Tremblay and Sorenson, "*An Introduction to Data Structures with Applications*",
3. McGraw hill, Kongakusha.
4. Horowitz Sahni Mehta, "*Fundamentals of Data structures*", SBCS Publication
5. Data Structures by Rajni Jindal.

Online Resources:

1. https://nptel.ac.in/courses/106102064

Course Code: **CSE211C**	Course Title: **Object Oriented Programming**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- To understand the principles of object-oriented programming with C++,
- To identify and practice the object-oriented programming concepts and techniques,
- To solve real-world problems through object-oriented approach,
- To understand functions, classes, data and objects in C++,
- To understand the concepts of operator overloading, virtual functions, polymorphism and inheritance with the understanding of early and late binding, and
- Understanding advanced features of C++ specifically strings and templates.

Unit – I (7)

Introduction: Basic features and concepts of Object-Oriented Programming (OOP), Benefits, Languages and Applications of OOPs.
Tokens, Expressions and Control Structures: Tokens, Keywords, Identifiers and Constants, Basic Data types, User-defined Data types, Derived Data Types, Memory Management Operators, Manipulators, Expressions, Operator Overloading, Control Structures.

Unit – II (10)

Classes and Objects: Specifying a class, defining member functions, private member functions, array within a class, memory allocation for objects, arrays of objects, objects as function arguments, returning objects, pointers to members, local classes, Nested Class, this pointer.
Constructors and Destructors: Constructors, Parameterized Constructors, Constructors with Default arguments, Dynamic Initialization of objects, Dynamic Constructors and Destructors, Recursive Constructor, Constructor and Destructor with Static Members

Unit – III (10)

Functions in C++: Main function, function prototyping, call by reference, Returning more values by Reference, inline functions, default functions, function overloading. Static data members, static function members.
Overloading and Type Conversion: Definition and Rules of overloading Operators, Overloading Binary and Unary Operators. Overloading operators using friends

Unit – IV (9)

Polymorphism and Inheritance: Polymorphism, Compile time polymorphism – function overloading and operator overloading. Pointers, Pointers to Objects and derived classes. Run time polymorphism – function overriding.
Inheritance – Types of inheritance, constructors and destructors in inheritance. Constraints of multiple inheritance. Abstract base class – virtual and pure virtual functions.

Unit – V (10)

Exception handling: Introduction, Principles of Exception Handling, The Keywords Try, Throw and Catch, Exception Handling Mechanism, Multiple Catch Statements, Catching Multiple Exceptions, Re-throwing Exception, Specifying Exception, Exceptions in Constructor and Destructors, Controlling Uncaught Exceptions.
Templates: Introduction, Function templates, Class templates, STL – Container, Algorithm, Iterator – vector, list, stack, map.

Textbooks:

1. Object-Oriented Programming in C++, Robert Lafore, Pearson Education India.
2. C++ Programming Language, Bjarne Stroustrup, Addison-Wesley.

Reference Books:

1. Effective Modern C++, Scott Meyers, Shroff/O'Reilly.
2. C++ Primer, Stanley Lippman, Addison-Wesley.
3. Object-Oriented Programming with C++, E Balagurusamy, Tata McGraw Hill.
4. The Complete reference – C++, Herbert Shield

Course Code: **CSE212C**	Course Title: **Database Management System**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- Identify the basic concepts and various data model used in Database design.
- Apply relational database theory and be able to describe relational algebra expression, tuple and domain relation expression for queries and SQL for implementing the queries.
- Recognize and identify the use of normalization and functional dependency.
- Apply and relate the concept of transaction, concurrency control in database.
- Apply recovery, indexing and hashing technique in database.

Unit – I (6)

Introduction: Introduction to database management, Characteristics of the Database, Database Systems, Data Models, data abstraction, and system structure, Purpose of database system, uses of database approach, database applications, Views of data, Database languages, Database system – Concepts and architecture, Database users and administrator, database types.

Unit – II (10)

Data models: definition and types, Entity-Relationship Model (E-R Model), E-R diagrams, entity set, relationship sets, mapping, cardinalities. Extended ER model. Translating ER Model into Relational Model. Introduction to relational databases, The relational model -Keys. Relational algebra, Tuple relational calculus.

Unit – III (10)

Database design: Relational database design, Functional dependencies, Non-loss decomposition, First, Second, Third Normal Forms, Dependency Preservation, Boyce/Codd Normal Form, Multi-Valued Dependencies and higher normal Forms.

Unit – IV (10)

Transactions: Transaction Concepts, ACID Properties, System Recovery, Concurrency, need for concurrency, locking protocols, Timestamp based Protocol, Deadlock, Serializability.

Unit – V (9)

Implementation techniques: Failures and their classification, recovery and atomicity, recovery algorithms, File organization, indexing (e.g., B and B+ trees).

Textbooks:

1. R. and Navathe, S.B., “Fundamentals of Database Systems”, Pearson Education.

Reference Book:

1. Database system Concept by Silberschatz and Korth.
2. Abraham, H. and Sudershan, S., “Database System Concepts”, McGraw-Hill.Elmasri.
3. Ramakrishnan, R. and Gekhre, J., “Database Management Systems”, Tata McGraw-Hill.

Course Code: **CSE213C**	Course Title: **Discrete Mathematics**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- To gain an understanding of the fundamental concepts in discrete mathematics.
- Understand the notion of mathematical thinking, mathematical proofs, and algorithmic thinking
- To become familiar with various areas of discrete mathematics, including set theory, combinatorics, graph theory, discrete structures, and advanced topics.
- To develop the skills to apply discrete mathematics to solve problems in computer science and related fields.
- To prepare for interdisciplinary research that combines mathematics, computer science, and other disciplines to solve problems and advance the field of discrete mathematics.

Unit – I (9)

Propositional Logic, Applications of Propositional Logic, Propositional Equivalences, Predicates and Quantifiers, First order logic, Nested Quantifiers, Rules of Inference

Unit – II (9)

Sets, Venn Diagrams, power set, Cartesian product, Set Notation, Set operations, Set Identities, Computer representation of sets, Boolean functions, Identities of Boolean Algebra, Duality, Abstract Definition of a Boolean Algebra, Sum of Products and Product of sums Expansion, Functional Completeness, NAND and NOR implementation.

Unit – III (9)

Functions, relations, reflexive, symmetric, antisymmetric, transitive, composition, Representing relations, Equivalence relations, partial orders and lattices, Monoids, Groups

Unit – IV (9)

Graph Terminology, Graphs: connectivity, matching, coloring, Handshaking Lemma, Konigsberg seven bridge problem, Euler graphs, Euler's theorem, Hamiltonian path and circuits, Graph coloring, chromatic number, isomorphism and Homomorphism of graphs, Trees terminology, properties of trees, Application of graphs and trees.

Unit – V (9)

Counting, recurrence relations, generating functions, Sum and product rule, Principle of Inclusion Exclusion. Pigeonhole Principle

Textbooks:

1. C. L. Liu: Elements of Discrete Mathematics, Tata Mc-Graw Hill.
2. Kolman, Busby and Ross: Discrete Mathematical Structures,.PHI
3. Narsingh Deo: Graph Theory with Applications to Engineering and Computer Sciences, PHI.
4. Murry R. Spiegel: Discrete Mathematics (Schaums Outline series), Tata McGraw Hill

Reference Books:

1. Kenneth H. Rosen: Discrete Mathematics and its applications,5th Ed. Tata McGrawHill
2. K.R Parthasarty: Basic Graph Theory, Tata Mc-Graw Hill

Course Code: **CSE214C**	Course Title: **Digital Electronics and Logic Design**	Credits: **03** L – **3** P – **0**

Course Outcomes (COs):

- To introduce the basic concepts of digital systems and the use of Boolean algebra in logic analysis and design.
- Understand the principles and methodology of digital logic design at the gate and switch level, including both combinational and sequential logic elements.
- To introduce basic tools of logic design and provide hands-on experience designing digital circuits and components through simple logic circuits to hardware description language
- Apply Boolean algebra and other techniques to express and simplify logic expressions.
- Analyze and design combinational and sequential digital systems.
- Use different techniques, among them a hardware description language and a programming language, to design digital systems.

Unit – I (8)

Introduction

Number Systems - Decimal, Binary, Hexadecimal, Octal Number systems and their Conversions, Arithmetic operations, subtraction using 1's and 2's complement, Binary coded decimal, Excess-3 Codes, Gray Codes, Binary weighted code.

Introduction to Boolean algebra and Boolean theorems, Minimization of Boolean Expressions by using theorems, Different types of Logic Gates and implementation of logic circuits using logic gates, Binary Arithmetic.

Unit – II (10)

Simplification of Boolean Expressions / Logic Gates

Introduction to minterms, maxterms, sum of product and product of sum representation of Boolean function, Simplification techniques and minimization by K-map method and Tabular (Q-M) method, NAND and NOR implementation.

Unit – III (10)

Combinational Logic Circuits

Design of various combinational circuits: Half/Full Adder and Subtractor, Ripple carry adder, Carry look ahead adder, Binary Adder/Subtractor, BCD adder, Binary Multiplier, Magnitude comparator, Multiplexers, De-Multiplexers, Decoders, Encoders.

Unit – IV (9)

Sequential Logic Circuits and Digital IC Families

Introduction to latches and flip flops, truth table and excitation table of flip flops, conversions of flip flops, design of synchronous and asynchronous counters, design of various types of shift registers.

Digital IC families: DTL, TTL, ECL, MOS, CMOS and their interfacing.

Unit – V (10)

Digital Circuit Design and Semiconductor Memories

Introduction to Moore and Mealy systems, state diagrams and tables, state reduction, design and analysis of Moore and Mealy systems.

Semiconductors Memories like ROM and RAM, Introduction to programmable logic design: PLA, PAL, ADC and DAC.

Textbooks:

1. Digital Logic and Computer Design, M. Morris Mano, Pearson Education India.
2. Digital Systems: Principles and Applications, Ronald J. Tocci, Neal Widmer, Greg Moss, Pearson Education India.

Reference Books:

1. Digital Design: With an Introduction to the Verilog HDL, M. Morris Mano, Michael D. Ciletti, Pearson.

Course Code: **CSE215C**	Course Title: **Data Structures Lab**	Credits: **01** L – **0** P – **2**

Course Outcomes (COs):

- Define, understand, describe and implement linear data structures like stack, queues, linked lists and nonlinear data structures like trees and graphs.
- Design and trace the algorithms for various operations on different data structures studied.
- Understand and implement various searching and sorting techniques
- Write programs in C to simulate operations and applications of data structures learnt.

List of Experiments

Units	Topics
1.	Program on arrays.
2.	Implementation of String Manipulation
3.	Program that uses functions to perform the following operations on singly linked list: (i) Creation (ii) Insertion (iii) Deletion (iv) Traversal.
4.	Program that uses functions to perform the following operations on doubly linked list: (i) Creation (ii) Insertion (iii) Deletion (iv) Traversal.
5.	Program that uses functions to perform the following operations on circular linked List: (i) Creation (ii) Insertion (iii) Deletion (iv) Traversal.
6	Program that implements stack (its operations) using: (i) Arrays (ii) Linked list (Pointers).
7.	Program to Implement array-based circular queue.
8.	Program that implements the following sorting: (i) Bubble sort (ii) Selection sort (iii) Quick sort (iv) Insertion sort (v) Merge sort (vi) Heap sort.
9.	Program to perform the following operations: (a) Insert an element into a binary search tree. (b) Delete an element from a binary search tree. (c) Search for a key element in a binary search tree.
10.	Program to perform the following operations: (a) Insert an element into an AVL tree. (b) Delete an element from an AVL tree. (c) Search for a key element in an AVL tree.
11.	Basic programs on implementation of graphs.

Course Code: **CSE216C**	Course Title: **Object Oriented Programming Lab**	Credits: **01** L – **0** P – **2**

Course Outcomes (COs):

- To identify and practice the object-oriented programming concepts and techniques,
- To solve real-world problems through object-oriented approach,
- Apply C++ features to program design and implementation.,
- Use C++ to demonstrate practical experience in developing object-oriented solutions,
- Analyze a problem description and design and build object-oriented software using good coding practices and techniques, and
- Use common software patterns in object-oriented design and recognize their applicability to other software development contexts.

List of Experiments

1. Design a class to represent a bank account. Include the following.
Data Members
- Name of the depositor
- Account number
- Type of account
- Balance amount in the account

Methods
- To assign initial values
- To deposit an amount
- To withdraw an amount after checking balance
- To display the name and balance

Incorporate a constructor to provide initial values.

2. Assume that a bank maintains two kinds of account for its customers, one called saving account and the other current account. The saving account provides compound interest and withdrawal facilities but no cheque book facility. The current account provides cheque book facility but no interest. Current account holders should also maintain a minimum balance and if it falls below a specified level, a service charge is imposed. Create a class *Account* that stores customer name, account number, and type of account. From this derive the classes *Curr-acct* and *Sav-acct* to make them more specific to their requirements. Include the necessary methods in order to achieve the following tasks.
- Accept deposit from a customer and update the balance
- Display the balance
- Compute and deposit interest
- Permit withdrawal and update the balance
- Check for the minimum balance, impose penalty, if necessary and update the balance.

Do not use any constructors. Use methods to initialize the class members.

3. Develop a program which will read a string and rewrite it in the alphabetical order. For example, the word *STRING* should be written as *GINRST.*

4. Design a generic function for finding the largest of three numbers.

5. An election is contested by 5 candidates. The candidates are numbered 1 to 5 and the voting is done by marking the candidate number on the ballot paper. Write a C++ program to read the ballots and count the votes cast for each candidate using an array. In case, a number read is outside the range 1 to 5, the ballot should be considered as a 'spoilt ballot' and the program should also count the number of spoilt ballots.

6. Create a class by name *date* with the member data as *day*, *month* and *year*.
Perform the following:
- Overload all relational operators <,<=,>,>=,==,=
- Overload ++ operator to increment a date by one day
- Overload + to add given number of days to find the next date
- Provide the necessary function to use the statement like days=dt; where days is an int variable and dt is an object of date class. The statement is intended to assign the number of days elapsed in the current year of the date to the variable days. Note that this is a case of conversion from derived type to basic type.

7. Design a class template by name Vector and perform the following:
- Find the smallest of the element in the Vector
- Search for an element in the Vector
- Find the average of the element in the array.

8. Exception Handling

Course Code: **CSE316C**	Course Title: **Database Management System Lab**	Credits: **01** L – **0** P – **2**

Course Outcomes (COs):

- Transform an information model into a relational database schema and to use a data definition language to implement the schema using a DBMS.
- Analyze the database using queries to retrieve records.
- Formulate queries using DDL, DML, DCL and TCL commands.
- Develop application programs using PL/SQL.
- Analyze front end tools to design forms, menus, etc. and establish back-end connectivity Develop solutions using database concepts for real-time requirements.

List of Experiments

1. Introduction to SQL, RDBMS: Visualizing the architecture of RDBMS, Different data types and its implementation.
2. SQL commands: Implementation of Creating and managing SQL tables, DDL (Data definition language): Implementation of Create, Alter, drop, rename, truncate, comment.
3. Implementing Operators in SQL: Comparison operators, Arithmetic operators.
4. Implementing Operators in SQL: Relational operators, Logical operators (AND, NOT, OR), Special operators (BETWEEN, IS NULL, EXISTS, IN, LIKE).
5. Implementing Aggregate Functions in SQL: SQL functions: (COUNT, MIN, SUM, MAX, AVERAGE, LIKE).
6. Manipulating Data in tables: Data manipulation Language: (Implementing Select, Insert, Update, Delete, merge.), Retrieval of data from the table: Implementing queries on single table.
7. Implementation of constraints: Not null, Primary Key, Unique, Check, Foreign key)
8. Combining Tables and execution of queries on such tables: Perform Join, inner join, outer join, natural join and subtypes of each.
9. Combining the tables using operations like Union, Intersect, minus and executing the queries.

MongoDB:

10. Introduction to NoSQL Databases.
11. MongoDB Ecosystem.
12. CRUD Operations:
    a) Create
    b) Read
    c) Update
    d) Delete

Course Code: **CSE217C**	Course Title: **Digital Electronics and Logic Design Lab**	Credits: **01** L – **0** P – **2**

Course Outcomes (COs):

- How digital values of analog signals are represented in different logic families.
- Description of truth tables for digital logic circuits.
- Gate level implementation of a number of circuits like multiplexers etc.
- Evaluation of number of constraints (parameters) of combinational and sequential logic circuits.
- To implement the logic circuits in the design of VLSI / IC circuits using hardware descriptive languages like VHDL, VERILOG etc.

List of Experiments

1. To verify the truth table of AND, OR and NOT gates.
2. Design NAND, NOR, XOR and X-NOR gates and also verify their truth tables.
3. Design Half Adder and verify its truth table.
4. Design Full Adder and verify its truth table.
5. Design a Binary Adder.
6. Design Half subtractor and verify its truth table
7. Design Full subtractor and verify its truth table.
8. Design multiplexer and demultiplexer using 2-input NAND gates.
9. Design Encoders and Decoders.
10. Realization of Flip-Flops (SR, JK, T, D flip-flops).

Course Code: **CSE210A**	Course Title: **Seminar**	Credits: **0** **Compulsory Audit**

Description:

In 3rd semester students have to take compulsory audit course titled ‘Seminar’. In this course they have to present a Power Point Presentation on a pre-approved topic from the department. The topic of the seminar should be chosen by keeping in view the **“current technological trends and advancements”** in computer science and engineering. A departmental committee will evaluate the individual students’ performance on the basis of their presentation and knowledge about the topic chosen for seminar.